Universidad Mariano Gálvez
Ingeniería en Sistemas de la Información
Automatas y Lenguajes Formales
Ing. Bryan Darren Orantes Muñoz

PROYECTO DE AUTOMATA - ANALIZADOR SINTÁCTICO

Integrantes:
6590-14-13194 Edward Anthony Salazar Rosales
6590-17-23948 Abraham Esaú Mazate Palacios
6590-14-1016 Pedro Luis Najera Kel

03 de octubre de 2020

INTRODUCCIÓN

El presente trabajo tiene como objetivo primordial, dar a conocer sobre las diversas funcionalidades de un analizador léxico, un analizador sintáctico, función de tokens uso de un alfabeto, gramática, y sus aplicativos a continuación se explica brevemente respecto a los temas mencionados.

Analizador Léxico:

El analizador léxico es la primera fase de un compilador, una de sus principales funciones consiste en leer los caracteres de entrada y elaborar como salida una secuencia de componentes léxicos que utiliza el analizador sintáctico para hacer el análisis. Esta interacción suele aplicarse convirtiendo al analizador léxico en una subrutina o co rutina del analizador sintáctico.

Analizador Sintáctico:

Su función es analizar una cadena o texto en componentes sintácticos lógicos, este puede interpretarse como un programa donde el compilador se asegura de que el código se traduzca correctamente a un lenguaje ejecutable. La tarea del analizador es, en este caso, la descomposición y transformación de las entradas en un formato utilizable para su posterior procesamiento.

Token:

Los tokens también pueden ser llamados componentes léxicos es una cadena de caracteres que tiene un significado coherente al interpretarse en algún lenguaje de programación.

Gramática:

Se define como un conjunto de reglas, que describen o analizan una secuencia de símbolos pertenecientes a un lenguaje en específico.

OBJETIVOS

1) Crear un analizador sintáctico y léxico que pueda interpretar tokens y a su vez, aplique una gramática interponiendo reglas y excepciones

2) El lenguaje de programación en el que se desarrolle el proyecto deberá ser capaz de dar a conocer las excepciones y palabras reservadas, las cuales deberá mostrar en pantalla.

3) El analizador será capaz de interpretar pedazos de código.

4) El analizador debe de interpretar y diferenciar terminales y no terminales.

5) El desarrollo de este proyecto tiene como objetivo primordial, dar a conocer los conocimientos adquiridos por los estudiantes durante el curso de autómatas y lenguajes formales.

CÓDIGO

A continuación se presentan los requerimientos para ejecutar el programa y el código del programa realizado en Visual Studio en el lenguaje C# (C Sharp).

Requerimientos del Sistema:

Windows 7, 10
Net Framework 4.7.X o en adelante
Mínimo 1 GB RAM
Mínimo espacio de 10 MB

Estructura

La programación en C# es top-down. La estructura a un nivel alto del código es:

Librerías
Clase
{
    Función Principal
    {
        Variables
        Procedimientos
    }
}

Librerías

```
//Librerias
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
```

Las librerías utilizadas fueron las que vienen por defecto en las plantillas o framework para la creación de una aplicación de consola .NET Framework. La única librería agregada por nosotros fue la librería “Collections.Generic”, ya que se utilizó una variable de tipo “lista”.

Clase

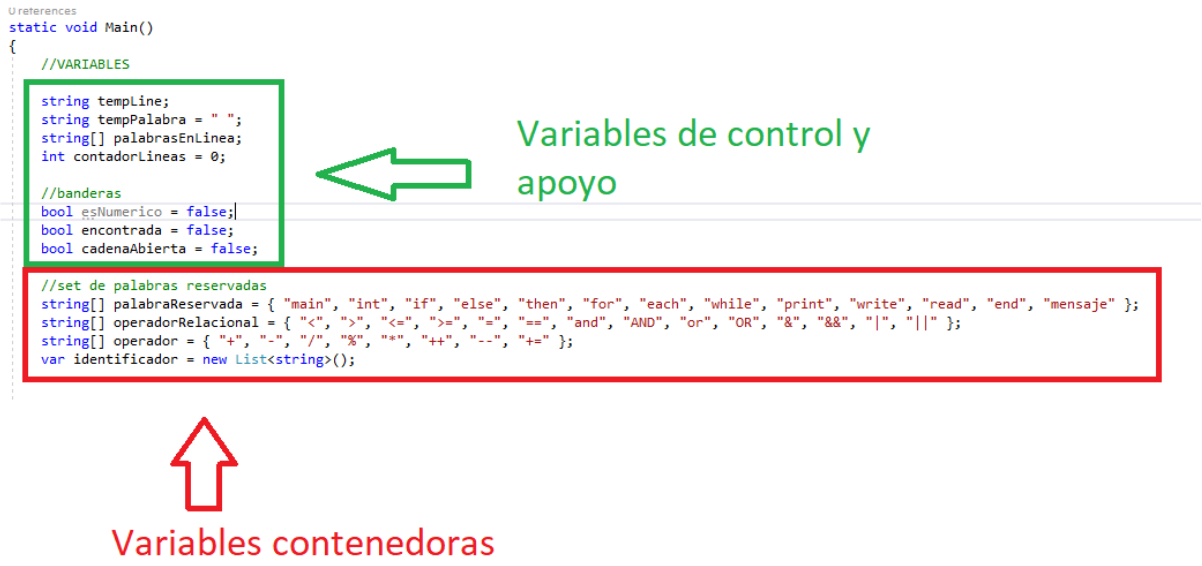
La clase creada fue “ReadFromFile” que se traduce a “LeerDeArchivo”, esto debido a que nuestro programa necesitara leer un de un archivo .txt.

```
using System.ComponentModel;
using System.Linq;

namespace ConsoleApp1
{
    0 references
    class ReadFromFile
    {
        0 references
        static void Main()
        {
```

Variables

Usamos variables y las dividimos en dos categorías: de <<control y apoyo>> y <<contenedoras>>. Las variables de control y apoyo sirven para poder manipular datos como asignar temporalmente valores o llevar un conteo. También, incluyen las “banderas”, que son variables booleanas que nos ayudan a saber si algún evento ya sucedió. Las variables contenedoras nos ayudan a poder contener valores que luego usaremos en el código para comparar y/o buscar. Por ejemplo, almacenamos todas las palabras consideras reservadas. Luego explicaremos a detalle como funciona el programa, pero a grandes rasgos el programa obtiene una palabra nueva y busca entre las palabras reservadas para ver si la palabra obtenida es una de ellas y poder clasificarlas.



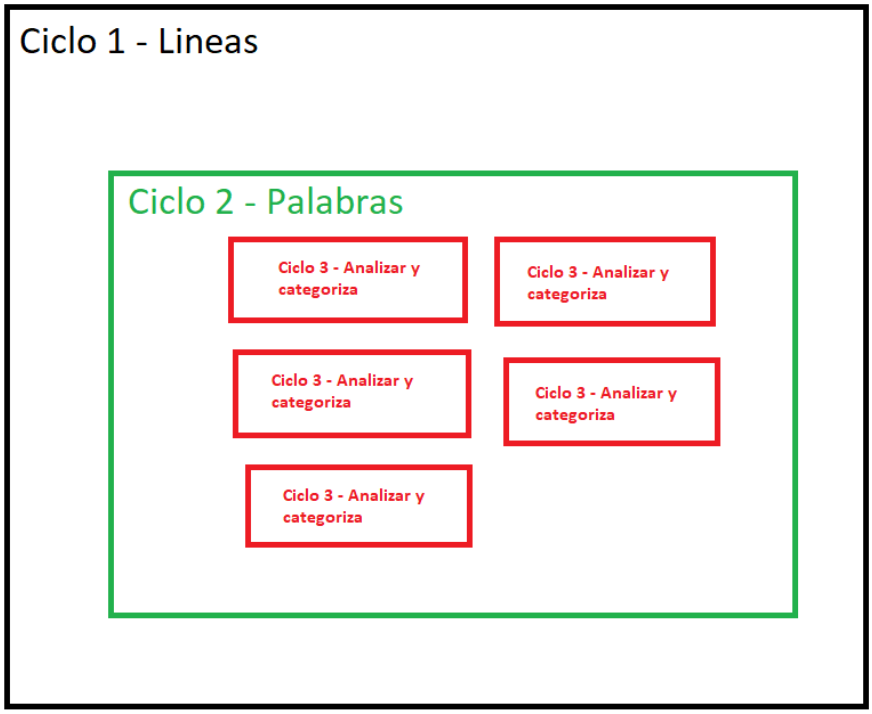
Try Catch

El programa “operativo” o funcional se encapsuló dentro de un Try-Catch. Esto se hizo por si hubiese un error fatal o de sistema al momento de ejecutar el programa no se cierre el programa de golpe y saber el porqué del error. El Catch fue programado para desplegar el mensaje de la excepción y así tener conocimiento de que hizo fallar el programa.

Una excepción de usuario seria que el usuario al momento de ejecutar el programa ingrese una ruta o path del archivo input invalida. Otra excepción seria que el archivo no sea .txt.

Programa

A continuación se detallará paso a paso como funciona el programa cronológicamente. Prácticamente, el programa consta de tres ciclos distintos. El ciclo 1 o de “lineas” busca obtener y formatear las líneas del archivo. El ciclo 2 o de “palabras” busca poder dividir las líneas en palabras y los ciclos 3 o de “clasificación” son los encargados de comparar y analizar cada palabra y determinar en que categoría o conjunto califican.



1. Solicita una ruta de input donde se encuentra el archivo .txt a leer.

```
//Solicitar path del archivo a leerse
Console.WriteLine("Ingrese ruta: ");
String ruta = Console.ReadLine();
```

2. Se lee el archivo y se cuentan las líneas totales.

```
//guarda temporalmente las lineas del archivo en un array de strings
string[] lines = System.IO.File.ReadAllLines(ruta);

// Recorrido para desplegar todas las lineas guardadas
System.Console.WriteLine("Abriendo archivo...");
System.Console.WriteLine("Achivo contiene " + lines.Length + " lineas en total.");
System.Console.WriteLine("CONTENIDO: ");
System.Console.WriteLine(" ");
```

3. Ciclo 1: Lee una línea y la almacena en un string de un arreglo. Se eliminan espacios en blanco o indentaciones al principio o final de cada línea.

```
foreach (string line in lines)
{
    //Elimina espacios al inicio y final de linea
    tempLine = line.TrimStart();
    tempLine = tempLine.TrimEnd();
    Console.WriteLine("Linea No." + contadorLineas);
    Console.WriteLine(tempLine);
    contadorLineas++;
    System.Console.WriteLine(" ");
    palabrasEnLinea = tempLine.Split(' ');
```

4. Ciclo 2: Luego que ya se tiene la línea, se busca dividir en palabras la linea. Esto se logra dividiendo la linea cada vez que se encuentra un espacio en blanco “\0”.

```
//parte cada linea en palabras
foreach (string palabra in palabrasEnLinea)
{
    //Analiza para ver si la palabra se encuentra entre las palabras y operadores reservados
    //esto se logra recorriendo los arreglos de palabras y operadores reservados
    //sino esta, verifica si es numero o cadena


    //si se abre una cadena con ", todo el texto que siga hasta que se encuentre " otra vez es considerado una cadena
    if(cadenaAbierta == false)
    {
        //revisa si la palabra esta en arreglo de palabraReservada
        foreach (string arraypalabra in palabraReservada)
        {
            if (palabra == arraypalabra)
            {
                Console.WriteLine(palabra + " es P RESERVADA encontrada en linea No." + contadorLineas);
                encontrada = true;
            }
        }
```

5. Ciclo 3: son los varios ciclos logrados por el recorrido de arreglos de palabras reservadas y operadores hechos con funciones “for each”. Prácticamente, obtenemos una palabra. Comparamos si esa palabra se encuentra en algún conjunto y si se encuentra se califica como la categoría en la que se encontró. Por ejemplo, si la palabra “mensaje” aparece en el archivo y también es parte de palabras reservadas, esta palabra será clasificada como “palabra reservada”.

```
//revisa si la palabra esta en arreglo de palabraReservada
foreach (string arraypalabra in palabraReservada)
{
    if (palabra == arraypalabra)
    {
        Console.WriteLine(palabra + " es P RESERVADA encontrada en linea No." + contadorLineas);
        encontrada = true;
    }
}
```

6. Al finalizar con la palabra en curso, se repiten los pasos 1-6 para la siguiente palabra hasta el final de línea. Luego se repiten los mismos pasos hasta terminar todas las líneas del archivo.

Sintaxis/Lógica

Como ya mencionamos, los ciclos 3 se encargan de hacer el análisis y clasificación de palabras. Para lograr esto, incorporamos las siguientes reglas en el código.

1. Palabras Reservadas: si el programa encuentra que la palabra analizada esta dentro del arreglo de palabras reservadas entonces la clasifica como tal.

2. Identificador: si el programa encuentra que la palabra analizada esta dentro del listado identificadores entonces la clasifica como tal. El identificador tiene una segunda regla, si una palabra no es reconocida la almacena temporalmente, luego si la siguiente palabra es un símbolo "=", entonces la palabra almacenada temporalmente o anterior se corrige y se agrega al listado de identificadores.
   Ejemplo:
   x = 5
   - x es palabra no reconocida.
   - x se almacena temporalmente.
   - Luego se encuentra el símbolo "=". Esto nos obliga a regresar a analizar "x".
   - Como se entiende que se encontró "x =" donde "x" se le asigna un valor, se corrige y "x" se categoriza y agrega a los identificadores.

3. Operador Relacional: similar a la regla 1, si el programa encuentra que la palabra analizada esta dentro del arreglo de operadores relacionales entonces la clasifica como tal.

4. Operador: similar a la regla 1, si el programa encuentra que la palabra analizada esta dentro del arreglo de operadores entonces la clasifica como tal.

5. Cadena: las cadenas se identifican al encontrar un símbolo de comillas dobles (") dos veces. Entonces, hay dos reglas. Al encontrar el primer set de comillas, la palabra se categoriza como cadena. La segunda, cualquier otra palabra que se encuentre antes de encontrar el segundo set de comillas es considerada cadena.
   Ejemplo:
   "Adiós x = 5" If else
   - Se encuentra el primer set de comillas (") por ende "Adiós" es considerado cadena.

- Se busca el segundo set de comillas pero en vez de eso se encuentra “x”, entonces “x” se considera cadena también. Lo mismo con “=” y “5”.
- Finalmente, se encuentra el segundo set de comillas después del “5”. Entonces, se analiza “If” pero como ya no se encontraron las comillas dobles, entonces “If” se analiza normalmente y se categoriza como reservada. Lo mismo con “else”.

6. Numérico: se intenta parsear o convertir la palabra al tipo INT (numérico entero). Si no hay error, se categoriza como numérica la palabra. Si da error o no se puede convertir es porque la palabra no continúe únicamente caracteres numéricos.

7. No reconocida: si la palabra no se ha categorizado con ninguna de las reglas anteriores, entonces, para entonces la palabra es categorizada como no reconocida.

RESULTADO DE PRUEBAS

Prueba 1

Ingrese ruta:
C:\Users\pedro\OneDrive\Documents\2. Universidad\Automatas\ALF_Analizador_CS_v1.0_Oct2020\ConsoleApp2\archivos de prueba\archivo1.txt
Abriendo archivo...
Achivo contiene 10 lineas en total.
CONTENIDO:

Linea No.0
int main

int es P RESERVADA encontrada en linea No.1

main es P RESERVADA encontrada en linea No.1

Linea No.1

Linea vacia en linea No.1

Linea No.2
int x = 5

int es P RESERVADA encontrada en linea No.3
x es palabra NO RECONOCIDA encontrada en linea No.3
CORRECCION: x es IDENTIFICADOR encontrada en linea No.3
= es OP RELACIONAL encontrada en linea No.3
5 es NUMERICA encontrada en linea No.3

Linea No.3

Linea vacia en linea No.3

Linea No.4
if x >= 7

if es P RESERVADA encontrada en linea No.5
x es IDENTIFICADOR encontrada en linea No.5
>= es OP RELACIONAL encontrada en linea No.5
7 es NUMERICA encontrada en linea No.5

Linea No.5
mensaje "vete de la vecindad"

mensaje es P RESERVADA encontrada en linea No.6
"vete es CADENA encontrada en linea No.6
de es CADENA encontrada en linea No.6
la es CADENA encontrada en linea No.6
vecindad" es CADENA encontrada en linea No.6

Linea No.6
else

else es P RESERVADA encontrada en linea No.7

Linea No.7
mensaje "no te vayas chavo"

mensaje es P RESERVADA encontrada en linea No.8
"no es CADENA encontrada en linea No.8
te es CADENA encontrada en linea No.8
vayas es CADENA encontrada en linea No.8
chavo" es CADENA encontrada en linea No.8

Linea No.8
end

end es P RESERVADA encontrada en linea No.9

Linea No.9

Linea vacia en linea No.9

Presione cualquier tecla para salir.

Prueba 2

Ingrese ruta:
C:\Users\pedro\OneDrive\Documents\2. Universidad\Automatas\ALF_Analizador_CS_v1.0_Oct2020\ConsoleApp2\archivos de prueba\archivo2.txt
Abriendo archivo...
Achivo contiene 5 lineas en total.
CONTENIDO:

Linea No.0
mensaje "hola como estas"

mensaje es P RESERVADA encontrada en linea No.1
"hola es CADENA encontrada en linea No.1
como es CADENA encontrada en linea No.1
estas" es CADENA encontrada en linea No.1

Linea No.1
mensaje "vete a dormir"

mensaje es P RESERVADA encontrada en linea No.2
"vete es CADENA encontrada en linea No.2
a es CADENA encontrada en linea No.2
dormir" es CADENA encontrada en linea No.2

Linea No.2
mensaje "x + 5"

mensaje es P RESERVADA encontrada en linea No.3
"x es CADENA encontrada en linea No.3

+ es CADENA encontrada en linea No.3
5" es CADENA encontrada en linea No.3

Linea No.3
mensaje "adios"

mensaje es P RESERVADA encontrada en linea No.4
"adios" es CADENA encontrada en linea No.4

Linea No.4
mensaje "end"

mensaje es CADENA encontrada en linea No.5
"end" es CADENA encontrada en linea No.5

Presione cualquier tecla para salir.

Prueba 3

Ingrese ruta:
C:\Users\pedro\OneDrive\Documents\2. Universidad\Automatas\ALF_Analizador_CS_v1.0_Oct2020\ConsoleApp2\archivos de prueba\archivo3.txt
Abriendo archivo...
Achivo contiene 10 lineas en total.
CONTENIDO:

Linea No.0
int main

int es P RESERVADA encontrada en linea No.1
main es P RESERVADA encontrada en linea No.1

Linea No.1

Linea vacia en linea No.1

Linea No.2
int x = 5

int es P RESERVADA encontrada en linea No.3
x es palabra NO RECONOCIDA encontrada en linea No.3
CORRECCION: x es IDENTIFICADOR encontrada en linea No.3
= es OP RELACIONAL encontrada en linea No.3
5 es NUMERICA encontrada en linea No.3

Linea No.3

Linea vacia en linea No.3

Linea No.4
if x == 8

if es P RESERVADA encontrada en linea No.5
x es IDENTIFICADOR encontrada en linea No.5
== es OP RELACIONAL encontrada en linea No.5
8 es NUMERICA encontrada en linea No.5

Linea No.5
mensaje vete a de la vecindad

mensaje es P RESERVADA encontrada en linea No.6
vete es palabra NO RECONOCIDA encontrada en linea No.6
a es palabra NO RECONOCIDA encontrada en linea No.6
de es palabra NO RECONOCIDA encontrada en linea No.6
la es palabra NO RECONOCIDA encontrada en linea No.6
vecindad es palabra NO RECONOCIDA encontrada en linea No.6

Linea No.6
else

else es P RESERVADA encontrada en linea No.7

Linea No.7
mensaje "no te vayas chavo

mensaje es P RESERVADA encontrada en linea No.8
"no es CADENA encontrada en linea No.8
te es CADENA encontrada en linea No.8
vayas es CADENA encontrada en linea No.8
chavo es CADENA encontrada en linea No.8

Linea No.8
end

end es CADENA encontrada en linea No.9

Linea No.9

es CADENA encontrada en linea No.10

Presione cualquier tecla para salir.

Prueba 4

Ingrese ruta:
C:\Users\pedro\OneDrive\Documents\2. Universidad\Automatas\ALF_Analizador_CS_v1.0_Oct2020\ConsoleApp2\archivos de prueba\archivo4.txt
Abriendo archivo...
Achivo contiene 5 lineas en total.
CONTENIDO:

Linea No.0
x = x + 5

x es palabra NO RECONOCIDA encontrada en linea No.1
CORRECCION: x es IDENTIFICADOR encontrada en linea No.1
= es OP RELACIONAL encontrada en linea No.1
x es IDENTIFICADOR encontrada en linea No.1
+ es OPERADOR encontrada en linea No.1
5 es NUMERICA encontrada en linea No.1

Linea No.1
if b > a

if es P RESERVADA encontrada en linea No.2
b es palabra NO RECONOCIDA encontrada en linea No.2
> es OP RELACIONAL encontrada en linea No.2
a es palabra NO RECONOCIDA encontrada en linea No.2

Linea No.2
a ++

a es palabra NO RECONOCIDA encontrada en linea No.3
++ es OPERADOR encontrada en linea No.3

Linea No.3
b --

b es palabra NO RECONOCIDA encontrada en linea No.4
-- es OPERADOR encontrada en linea No.4

Linea No.4
x+2+24

x+2+24 es palabra NO RECONOCIDA encontrada en linea No.5

Presione cualquier tecla para salir.

Prueba 5

Ingrese ruta:
C:\Users\pedro\OneDrive\Documents\2. Universidad\Automatas\ALF_Analizador_CS_v1.0_Oct2020\ConsoleApp2\archivos de prueba\archivo5.txt
Abriendo archivo...
Achivo contiene 8 lineas en total.
CONTENIDO:

Linea No.0
hola este es un mensaje

hola es palabra NO RECONOCIDA encontrada en linea No.1
este es palabra NO RECONOCIDA encontrada en linea No.1
es es palabra NO RECONOCIDA encontrada en linea No.1
un es palabra NO RECONOCIDA encontrada en linea No.1
mensaje es P RESERVADA encontrada en linea No.1

Linea No.1
que mezcla palabras reservadas como

que es palabra NO RECONOCIDA encontrada en linea No.2
mezcla es palabra NO RECONOCIDA encontrada en linea No.2
palabras es palabra NO RECONOCIDA encontrada en linea No.2
reservadas es palabra NO RECONOCIDA encontrada en linea No.2
como es palabra NO RECONOCIDA encontrada en linea No.2

Linea No.2

Linea vacia en linea No.2

Linea No.3

Linea vacia en linea No.3

Linea No.4
mensaje

mensaje es P RESERVADA encontrada en linea No.5

Linea No.5
end if

end es P RESERVADA encontrada en linea No.6
if es P RESERVADA encontrada en linea No.6

Linea No.6
y tambien operadores como

y es palabra NO RECONOCIDA encontrada en linea No.7
tambien es palabra NO RECONOCIDA encontrada en linea No.7
operadores es palabra NO RECONOCIDA encontrada en linea No.7
como es palabra NO RECONOCIDA encontrada en linea No.7

Linea No.7
+ > = <=

+ es OPERADOR encontrada en linea No.8
> es OP RELACIONAL encontrada en linea No.8
CORRECCION: como es IDENTIFICADOR encontrada en linea No.8
= es OP RELACIONAL encontrada en linea No.8
<= es OP RELACIONAL encontrada en linea No.8

Presione cualquier tecla para salir.

Excepciones

1. El programa es “case sensitive”, es decir, no es lo mismo “mensaje” que “MENSAJE”. Las letras mayúsculas y minúsculas no son lo mismo.

2. Cada palabra debe estar separada por un espacio en blanco. De lo contrario, si no hay un espacio de por medio entre palabras, se considera una sola palabra.

3. El programa está configurado para leer archivos de texto .txt. El programa podría leer otros formatos o extensiones pero podría tener resultados alternos a los esperados.
4. El programa buscará cerrarse si se le ingresa una ruta o path invalido.

5. El programa buscará cerrarse por cualquier error de sistema o excepción antes mencionada.

ASIGNACIONES DE TAREAS

Durante la **semana 1** se hizo la carátula, introducción, objetivos, los 10 códigos de prueba, y también hicimos una tabla en Excel con todas las palabras reservadas, operadores relacionales dígitos y operadores aritméticos. La introducción, objetivos fueron hechas por Abraham Mazate, los 10 códigos de prueba fueron hechos por Pedro Nájera y la carátula y el Excel con todas las palabras reservadas, operadores relacionales dígitos y operadores aritméticos fueron hechas por Edward Salazar.

Durante la **semana 2** analizamos que tipo de lenguaje utilizamos e investigamos acerca de librerías, parsers, analizadores léxicos y las posibilidades de hacer el código. La investigación se hizo por cada integrante del grupo(Abraham, Pedro, Edward), ya que todos teníamos que estar en la misma página. Luego nos reunimos todos (Abraham, Pedro, Edward) en Zoom para la discusión. Cabe mencionar que no tomamos en cuenta Java por la complejidad, accedimos a utilizar C + + por una librería de Bison y Flex.

Durante la **semana 3** se inició la implementación del proyecto en C + +. Durante esta semana se hizo la estructura del código con strings y tokens. Abraham hizo la estructura del menú en el archivo C + +, Edward hizo la estructura de la apertura del archivo y el cerrado del archivo.

Durante la semana todos nos reunimos al menos 1 vez en Zoom para poder apoyar y continuar el proyecto. Si alguien tenía dudas creamos un grupo de WhatsApp para resolver dudas y presentar avances. Creamos una carpeta de drive donde almacenamos todos los archivos utilizados y agregamos las versiones/prototipos trabajadas. Cabe mencionar que Abraham hizo una consulta hacia su persona porque aún teníamos duda de que nos faltaba.

Durante la **semana 4** no se trabajó dado a los exámenes parciales.

Durante la **semana 5** nos percatamos de que no podíamos obviar los espacios en C + +. Sin embargo para remediar esto se trabajó en C# por una función llamada TRIM. Tuvimos que trabajar desde 0. Sin embargo, visualizamos que C# tiene más recursos que C + +. Entonces, nuestro proyecto final lo hicimos con C#. Cabe mencionar que para tener el proyecto listo el equipo completo puso de su parte y nos esforzamos para terminarlo a tiempo. En esta semana también se hizo la reportería. Las conclusiones, el glosario, maquillamos el código, índice y el resultado de las pruebas. Para esto nos dividimos el trabajo, Abraham hizo las conclusiones, Pedro hizo la parte del código y resultados de pruebas, y Edward hizo el glosario y asignación de tareas.

CONCLUSION

Después de realizar este proyecto podemos entender con más facilidad que todo proceso de clasificación conlleva el seguir reglas o condiciones. En un lenguaje específico, el escribir o formar palabras de manera correcta significa el aplicar una sintaxis. El proceso inverso de escribir, es decir leer y entender. Por ende, es importante entender la sintaxis de un lenguaje para poder leer y escribir bien.

Dado que nuestra carrera posee una rama que implica escribir de manera lógica y correcta, programar, es importante entender estos conceptos. No solo se aplicarán en programas de computadoras, sino también en aparatos o sistemas de la vida diaria y profesional.

GLOSARIO

Analizador léxico:
Un analizador léxico o también conocido como analizador lexicográfico es un programa que recibe como entrada el código fuente de otro programa en otras palabras una secuencia de caracteres y su salida está compuesta de tokens o símbolos.

Analizador sintáctico:
Un analizador sintáctico o también conocido como parser es un programa que asegura que se descompongan las entradas y también en transformar las entradas para poder ser usadas posteriormente.

C++:
Es un lenguaje de programación creado por Bjarne Stroustrp en 1979. Su intención fue extender el lenguaje C para poder manipular objetos.

Java:
Es un lenguaje de programación creado por James Gosling en 1995. Su sintaxis es derivada de C y C + +. Hay muchas aplicaciones que funcionan con Java.

C#:
Es un lenguaje de programación creado por la empresa Microsoft como parte de su plataforma. La sintaxis básica es derivada de C y C + +.

Bison:
Bison es un generador de analizadores sintácticos de propósito general que convierte una descripción para una gramática independiente del contexto.

Flex:
Flex es una herramienta que permite generar analizadores léxicos a partir de un conjunto de expresiones regulares.

Alfabeto:

Es un conjunto de símbolos finitos no vacíos.

Gramática:
Es un conjunto finito de reglas que describen toda la secuencia de símbolos pertenecientes a un lenguaje en específico.

Visual Studio:
Es un entorno de desarrollo integrado para Windows y macOS. En otras palabras es un IDE.

Cadena:
Es una secuencia finita de símbolos que pertenecen a un alfabeto.

Lenguaje de Programación:
Es un lenguaje formal(o artificial) que le proporciona a una persona en este caso programar.

Librerías:
Es un archivo o conjunto de archivos que se utilizan para facilitar la programación.

.NET Framework:
Es un framework desarrollado por Microsoft que principalmente corre en Microsoft Windows.

Variable:
Variable es una palabra que representa aquello que está sujeto a cambios.

Clase:
Es una plantilla para la creación de objetos de datos según un modelo predefinido.

Compilador:
Es un tipo de traductor que transforma un programa entero de un lenguaje de programación a otro.